

REMARKS

The claims remaining in the present application are Claims 16-35. Claims 33-35 have been added. No new matter has been added as a result of these amendments.

REJECTIONS

Claims 16-32 are rejected under 35 U.S.C. §102(b) as being anticipated by Kelly et al., U.S. Patent No. 5,832,205 (hereinafter, Kelly). The rejection is respectfully traversed for the following reasons. Applicants respectfully assert that Claims 16-32 are neither taught nor suggested by Kelly.

Claim 16 recites:

A method of determining validity of a translated instruction comprising:

- a) starting execution of a first host instruction translated from a first target instruction, wherein said first host instruction is linked from a second host instruction translated from a second target instruction, and wherein a first condition of a target system state required by said first host instruction holds;
- b) testing a second condition of said target system state to determine the validity of said first host instruction;
- c) executing said first host instruction if said second condition holds; and
- d) generating an exception if said second condition does not hold.

Claim 16 recites that a second condition of the target system state is tested to determine the validity of a first host instruction. Applicants respectfully submit that Kelly fails to teach or suggest this claimed limitation.

The rejection cites the "T-bit" as meeting this claimed limitation. The rejection appears to argue that testing the T-bit indicates that a valid host instruction exists somewhere. The rejection appears to conclude that this knowledge that a valid host instruction exists somewhere teaches the claimed limitation of testing a second condition of said target system state to determine the validity of said first host instruction.

However, Applicants respectfully assert that even if one concludes that testing the T-bit indicates that a valid instruction exists, it does not indicate that a particular host instruction is valid, which is what is claimed. Thus, Applicants do not understand the "T-bit" to be used to determine the validity of a first host instruction, as claimed. Rather, the T-bit is used to mark pages for which valid translations exist, such that target memory is not overwritten. Thus, the T-bit is not used to determine validity of a host instruction as claimed, but rather is used to prevent target memory from being overwritten.

The translation bit (T-bit) is used to indicate target memory pages for which translations exist. The T-bit thus possibly indicates that particular pages of target memory contain target instructions for which host translations exist. If an attempt is made to write to the protected pages in memory, the presence of the T-bit will cause an exception which when handled by the code morphing software can cause the appropriate translation(s) to be invalidated or removed from the translation buffer. The T-bit can also be used to mark other target pages that translation may rely upon not being written. (Col. 22, lines 35-48, emphasis added).

Further, Applicants note that the T-bit is also used to mark pages that a translation may rely upon not being written, as opposed to marking pages for which a translation exists (col. 22, lines 46-48). Thus, in cases such as this, if the T-bit for such a memory address is tested, the T-bit would indicate that the memory address should not be written. However, in this case, there are not any valid host instructions for this particular memory address (e.g., memory page). Hence, a false positive would result. Therefore, testing the T-bit is not an effective way to determine the validity of a first host instruction.

Prior to executing the translation, the T bit for the target page(s) containing the target instructions that have been translated is set. This indication warns that the instruction has been translated; and, if an attempt to write to the target address occurs, the attempt generates an exception which causes the translation to possibly be invalidated or removed. (Col. 23, lines 29-35, emphasis added).

For the foregoing rationale, Claim 16 is neither taught nor suggested by Kelly. As such, allowance of Claim 16 is respectfully requested.

Claim 24 recites:

A method of determining validity of a translated instruction comprising:

- a) performing a first address consistency check of a first host instruction made from a first target instruction to verify that said first host instruction is valid;
- b) executing said first host instruction;

c) determining whether a second host instruction made from a second target instruction and that is linked from said first host instruction can be safely executed without a second address consistency check; and

d) executing said second host instruction without performing said second address consistency check if safe.

Claim 24 recites “performing a first address consistency check of a first host instruction made from a first target instruction to verify that said first host instruction is valid.” For at least the reasons provided in the response to Claim 16, this claimed limitation of Claim 24 is neither taught nor suggested by Kelly.

Further, Applicants note that the second host instruction made from a second target instruction *is linked* to a first host instruction made from a first target instruction. Thus, the host instructions are *linked* and made from respective *target instructions*. Applicants note that not all host instructions that follow one another are linked to one another, as defined by Applicant.

Embodiments of the present invention involve linked instructions. When instructions are linked, rather than determining the next instruction to be executed by the target instruction EIP value, the next host instruction is determined by a jump command to the next translation placed at the end of the executing translation by a linking process of the code morphing software (Specification, page 5, lines 15-20). Thus,

Applicants have clearly defined that linking host instructions involves a jump command to the next translation placed at the end of the executing translation. The mere fact that two instructions follow one another does not make them linked, in accordance with Applicants' usage, as claimed.

The rejection asserts that the commit operation is used to determine whether a second host instruction can be safely executed without an address consistency check. The rejection has referred to Kelly at column 17, lines 1-39 and Figure 11. Applicants respectfully request that the Examiner point out where the first and second *linked* host instructions made from target instructions are in the cited passage and/or Figure 11 or withdraw this assertion.

For the following reasons, Applicants respectfully assert that Kelly fails to teach or suggest the limitations of Claims 24. Claim 24 recites, "determining whether a second host instruction made from a second target instruction and that is linked from said first host instruction can be safely executed without a second address consistency check." The rejection cites Kelly col. 17, lines 1-39 as meeting this claimed limitation. Applicants respectfully submit that this passage fails to teach or suggest this claimed limitation. In order to maintain a rejection under 35 U.S.C. §102, all limitations of the claim must be taught in the prior art reference, arraigned as in the claim.

Anticipation requires the presence in a single prior art reference disclosure of each and every element of the claimed invention, arranged as in the claim (*Lindemann Maschinefabrik GmbH v. American Hoist & Derrick Co.*, 221 USPQ 481, 485 (Fed. Cir. 1984)).

In contrast, to meeting the claimed limitation, the cited passage is concerned with the operation of the gated store buffer, which controls the transfer of data to memory. The passage indicates that memory stores generated during execution of host instructions are placed in the store buffer. Upon the occurrence of a commit operation, the memory stores generated during the execution are moved past the gate (committed) and written to memory. However, Applicants do not understand this passage to teach or suggest the claimed limitations of “determining whether a second host instruction made from a second target instruction and that is linked from said first host instruction can be safely executed without a second address consistency check.”

For the foregoing rationale, Claim 24 is neither taught nor suggested by Kelly. As such, allowance of Claim 24 is respectfully requested.

Claim 29 recites

A method of linking translated instructions comprising:

- a) translating a first target instruction to a first host instruction;
- b) determining that said first host instruction is to be linked to a second host instruction translated from a second target instruction; and
- c) providing an address consistency check for said first host instruction.

The rejection appears to consider the claimed limitations of “determining that said first host instruction is to be linked to a second host instruction translated from a second target instruction” to be taught by two instructions following one another in a sequence in the code example in column 25. That is, Applicants understand the rejection to interpret “linked instructions” as any two instructions that follow one another.

The embodiment recited in Claim 29 involves linked instructions. When instructions are linked, rather than determining the next instruction to be executed by the target instruction EIP value, the next host instruction is determined by a jump command to the next translation placed at the end of the executing translation by a linking process of the code morphing software (Specification, page 5, lines 15-20).

Applicants note that Kelly does provide an example of a sequence of instructions that finish with a link to another instruction at column 38, lines 12-22. In this example, the sequence of host instructions end with a jump instruction that points to a jump address furnished by chaining to another sequence of translated instructions (Kelly, col. 38, lines 24-31).

However, the code segment that the rejection cites at col. 25 of Kelly is not an example of linked instructions, as claimed. This is evident from the jump command at the end of the sequence in column 25 that returns to the main loop. By returning to the main loop, the sequence of instructions is not linked to another sequence of instructions. Moreover, contrary to the assertion in the rejection, two consecutive host instructions in the sequence of instructions in column 25 are not linked instructions. This is because they do not meet the Applicants stated condition of linked instructions in which the next host instruction is determined by a jump command to the next translation (Specification, page 5, lines 15-20).

Applicants further argue that the limitations of “providing an address consistency check for said first host instruction,” overcome the cited reference. The Applicants have claimed that the address consistency check is for the first host instruction, which the Applicants have claimed is to be linked to the second host instruction. The rejection cites the “chkl” and “chku” instructions in the example in col. 26 of Kelly as teaching this claimed limitation. However, the “chkl” and “chku” instructions are not used as address consistency checks for a host instruction that are linked, as claimed.

For the foregoing rationale, Claim 29 is neither taught nor suggested by Kelly. As such, allowance of Claim 29 is respectfully requested.

Claims 17-23, 25-28, and 30-32 depend from Claims 16, 24, and 29, which are believed to be allowable for the foregoing rationale. As such, Claims 17-23, 25-28, and 30-32 are believed to be allowable.

Claims 27-28 and 30-32 are respectfully believed to be patentable for the following additional rationale.

Claims 27-28 recite, in part:

wherein multiple target instructions are translated to host instructions and wherein said c) comprises determining whether any of said target instructions reside on different pages of memory.

Claim 27 and 28 further limit the limitation in Claim 24 of, "determining whether a second host instruction made from a second target instruction and that is linked from said first host instruction can be safely executed without a second address consistency check." Thus, the claimed, "determining whether any of said target instructions reside on different pages of memory," is a further limitation of "determining whether a second host instruction made from a second target instruction and that is linked from said first host instruction can be safely executed without a second address consistency check."

The rejection cites the compare operation as depicted in Figure 11 of Kelly as purportedly teaching determining whether any target pages reside on different pages of memory. Applicants respectfully assert that even if for the sake of argument it is assumed that Figure 11 does teach the individual concept of determining whether any target pages reside on different pages of memory, this does not teach or suggest the claimed limitation. This is because Claim 28, as a whole, recites an arrangement that is not taught nor suggested by Kelly. A part of the claimed arrangement is the claimed limitations of, "determining whether any of said target instructions reside on different pages of memory as further limitations of "determining whether a second host instruction made from a second target instruction and that is linked from said first host instruction can be safely executed without a second address consistency check."

For the foregoing reasons, Applicants respectfully assert that Claims 27-28 are patentable over Kelly.

Claim 30 recites, in part:

- c1) linking said second host instruction to said first host instruction; and
- c2) including code for performing said address consistency check as a part of said first host instruction

The rejection cites consecutive instructions in Kelly's code example on column 26 as "linked instructions." For reasons already discussed herein, the first and second "mov" instructions in Kelly's example are not linked instructions. Therefore, the limitation in c2) of "including code for performing said address consistency check as a part of said first host instruction," is not taught or suggested by Kelly as the "chk1" and "chku" instructions in Kelly are irrelevant to the claimed limitations as they are not included as a part of a first instruction, which is linked to a second instruction, as claimed.

Claim 31 recites, in part:

- c1) linking said second host instruction to code for performing said address consistency check; and
- c2) linking said code for performing said address consistency check to said first host instruction.

Claim 32 recites, in part:

- c1) linking said second host instruction to said first host instruction; and
- c2) incorporating code for performing said address consistency check into said first host instruction

For the reasons discussed in the response to Claim 30, Claim 31 and 32 is neither taught nor suggested by Kelly.

NEW CLAIMS

Claims 33-35 have been added. Support for Claims 33-35 may be found in the instant specification at least at page 8, lines 13-19.

Claim 33 recites, in part:

wherein said first condition is that an address stored in said second host instruction matches a physical address of said second target instruction.

Applicants respectfully assert that Kelly fails to teach or suggest this claimed limitation. The rejection has asserted that the condition of the A/N bit teaches the limitation of the "address consistency check," in Claim 17, which Claim 33 further limits. Applicants respectfully assert that Kelly's A/N bit cannot be used to match an address stored in a host instruction with a physical address, as the A/N bit is a single bit.

Claims 34 and 35 contain similar limitations as Claim 33 and are respectfully believed to be allowable for at the same reasons.


CONCLUSION

In light of the above listed amendments and remarks, reconsideration of the rejected Claims is requested. Based on the amendments and arguments presented above, it is respectfully submitted that Claims 16-35 overcome the rejections of record and, therefore, allowance of Claims 16-35 is earnestly solicited.

Should the Examiner have a question regarding the instant amendment and response, the Applicants invite the Examiner to contact the Applicants' undersigned representative at the below listed telephone number.

Dated: 6/2, 2004

Respectfully submitted,
WAGNER, MURABITO & HAO LLP


Ronald M. Pomerence
Registration No. 43,009

Address: WAGNER, MURABITO & HAO LLP
Two North Market Street
Third Floor
San Jose, California 95113

Telephone: (408) 938-9060 Voice
(408) 938-9069 FAX